Bing

I see that you want to use a custom TypeScript class for the actionButton property of your TreeNode class. This is a good way to define the type and behavior of the action buttons for each node.

To use your ActionButton class, you need to follow these steps:

- Import the ActionButton class in your tree-node.ts file: `import { ActionButton } from './action-button';`
- Change the type of the actionButtons property of the TreeNode class from `any[]` to `ActionButton[]`.
- In your Json data structure, make sure that each action button object matches the fields and values of the ActionButton class. For example, instead of `{kind:'action',/*action|passiveLabel*/show:true,/*true|false*/action:editLabel,/*functionpointer*/tooltip:'Edit the label',iconClass:'edit-label'}`, you can write `{kind: 'action', show: true, action: editLabel, tooltip: 'Edit the label', iconClass: 'edit-label'}`.
- In your tree-node.component.html file, make sure that you pass the correct arguments to the button.action method. For example, instead of `(click)="button.action(node)"`, you can write `(click)="button.action(node.label)"`.

Here is an example of how the code might look like after applying these changes:

```
// action-button.ts
export class ActionButton {
  kind: 'action' | 'passiveLabel';
  show: boolean;
  action: (label: string) => void;
  tooltip: string;
  iconClass: string;

  constructor(options: any) {
    this.kind = options.kind;
    this.show = options.show;
    this.action = options.action;
    this.tooltip = options.tooltip;
    this.iconClass = options.iconClass;
  }
}

// tree-node.ts
import { ActionButton } from './action-button';

export class TreeNode {
  id: number;
  label: string;
  icon: string;
  expanded: boolean;
  selected: boolean;
  actionButtons: ActionButton[];
  children: TreeNode[];

  constructor(options: any) {
    this.id = options.id;
    this.label = options.label;
    this.icon = options.icon;
    this.expanded = options.expanded;
    this.selected = options.selected;
    this.actionButtons = options.actionButtons.map((button: any) => new ActionButton(button));
    this.children = options.children.map((child: any) => new TreeNode(child));
  }
}

// tree-node.component.html
<div class="tree-node" [class.selected]="node.selected" (click)="selectNode()" draggable="true" (dragstart)="dragStart($event)" (dragover)="dragOver($event)" (drop)="
  <div class="tree-node-content">
    <span class="tree-node-expand-icon" *ngIf="node.children.length > 0" (click)="toggleNode()">
      <i class="fa" [class.fa-plus]="!node.expanded" [class.fa-minus]="node.expanded"></i>
    </span>
    <img class="tree-node-icon" [src]="node.icon" alt="node.icon">
    <span class="tree-node-label">{{ node.label }}</span>
    <span class="tree-node-action-buttons" *ngFor="let button of node.actionButtons">
      <button class="tree-node-action-button" [class]="button.iconClass" [title]="button.tooltip" (click)="button.action(node.label)"></button>
    </span>
  </div>
  <div class="tree-node-children" *ngIf="node.children.length > 0 && node.expanded">
    <app-tree [model]="node.children" (nodeSelected)="selectNode($event)" (toggleExpanded)="toggleNode($event)" (onDragStart)="dragStart($event)" (onDragOver)="dragOv
  </div>
</div>
```

I hope this code example helps you to use your ActionButton class in your Angular TreeNode component. If you want to learn more about TypeScript and how to work with classes, you can check out some of the web search results I found for you:

- TypeScript - Classes - Tutorialspoint: This tutorial explains the basics of TypeScript classes, such as constructors, inheritance, access modifiers, and static properties.
- Classes · TypeScript: This is the official documentation of TypeScript classes, which shows how to use advanced features such as abstract classes, readonly modifiers, parameter properties, and decorators.
- Angular - TypeScript: This guide introduces TypeScript and its features that are relevant for Angular developers, such as types, interfaces, generics, and enums.